09/865,243 PTO-892

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
30 November 2000 (30.11.2000)

PCT

(10) International Publication Number
WO 00/72200 A1

(51) International Patent Classification[7]: G06F 17/30

(21) International Application Number: PCT/US00/14055

(22) International Filing Date: 18 May 2000 (18.05.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/315,802 21 May 1999 (21.05.1999) US

(71) Applicant: INFRAWORKS CORPORATION [US/US]; 504 Lavaca Street, 11th floor, Austin, TX 78701 (US).

(72) Inventors: FRIEDMAN, George; 7109 Montana Norte, Austin, TX 78731 (US). STAREK, Robert, Phillip; 1779 Wells Branch Parkway, 110-B-284, Austin, TX 78728 (US). MOORMAN, Michael, J.; 1800 Parkside Lane, Austin, TX 78745 (US).

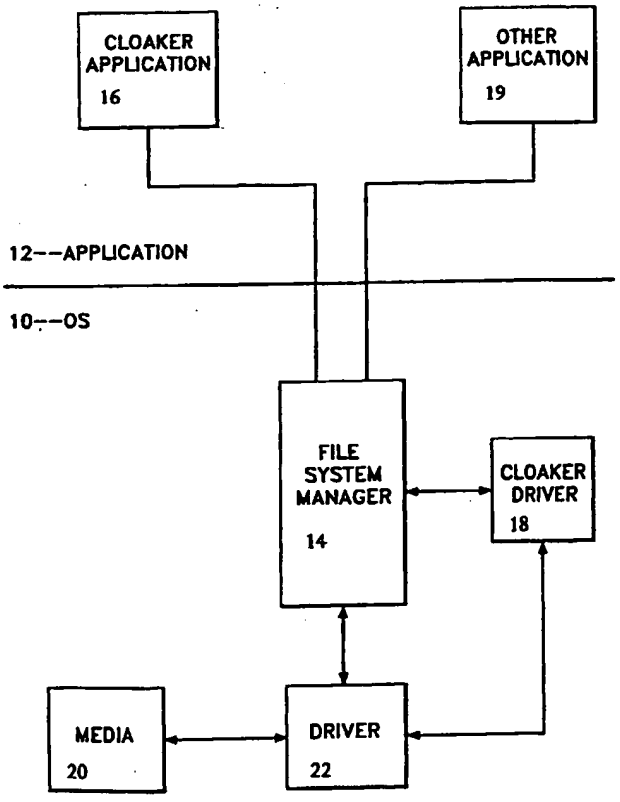
(74) Agents: TAUFER, Paul, A. et al.; Schnader Harrison Segal & Lewis, LLP, Suite 3600, 1600 Market Street, Philadelphia, PA 19103 (US).

(81) Designated States *(national)*: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European

[Continued on next page]

(54) Title: METHOD AND APPARATUS FOR SECURING FILES



(57) Abstract: A method and apparatus are provided for securing computer files by cloaking or hiding files from the user. A virtual device driver (22) monitors calls from and returns to the file manager to identify any references to files which have been previously stored in the drivers (22) database or lookup table, the driver (22) modifies the calls or returns effectively to render the file invisible. The user thus is not able to open the file or manipulate it in any way, and is not able to see the file in a directory listing. Thus, the user is not able to learn of the existence of the cloaked file (16). Selection of files for storage in the driver (22) database or lookup table is password protected to ensure that the authorized user may de-select and de-cloak the files at a later time as desired. The method and apparatus also permits restrictive use of files by monitoring the type of process identified in the call and by the type of action or procedure within the process that is to be performed. Thus, only selected application programs (19) may access a file stored in the driver (22) database and only certain procedures within the selected applications (19) may be performed. In one example, a word processing program may have access to a file (12) but may not be permitted to perform a delete function.

WO 00/72200 A1

patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Published:
— *With international search report.*

METHOD AND APPARATUS FOR SECURING FILES

TECHNICAL FIELD OF THE INVENTION

The present invention relates in general to the field of electronic systems, and more specifically to a method and apparatus for securing computer files.

BACKGROUND OF THE INVENTION

Security concerns as they pertain to computer files have gained heightened interest with the every increasing pervasiveness of computers in all phases of business and personal life. Employees and competitors have opportunities to examine files that contain sensitive business and financial data sometimes on the premises of the business and sometimes via internet or intranet access. Laptop computers are traveling everywhere. Business travelers carry sensitive company information in their laptop computers. Individuals taking personal holidays often take their laptops to do some work or keep in touch via e-mail while vacationing. Loss of such computers is a common mishap. Computer theft is also on the rise and more often than not the

value of the booty is less in the value of the hardware than in the value of the confidential business and personal information contained in the computer's memory (i.e., the "C" drive).

While encryption packages are useful in increasing security of files, it is advantageous to have additional or replacement security devices to limit file access and enhance security.

Generally speaking, there are three levels of security. At the highest level of security, it is desirable that the user neither have access to the sensitive information nor even know of its existence. At the next lower level of security, the user is permitted to know of the existence of certain information but is denied access to that information. At an even lower level of security, the user may be permitted to gain access to information but only via certain programs and under certain procedures.

None of the above described security features are currently possible in the Microsoft world. While both Windows 95/98 and NT 4.0 allows the users to render files technically invisible or non-manipulative, i.e., placed in a "write only" mode, these safeguards can be easily defeated by even the casual user. A robust, flexible system for providing increased levels of security for files is needed.

SUMMARY OF THE INVENTION

In accordance with the principles of the invention, a method and apparatus is provided for providing various levels of security. In the highest level of security, the user of a computer is not even able to see if a hidden or "cloaked" file exist on the system and thus, *a*

fortiori, is not able to access such file. In this embodiment, the apparatus of the invention provides for a virtual cloaker driver which is operative with the computer's file system to prevent the file rrom being opened, deleted, renamed and even listed in the file directory.

In another embodiment of the invention, the user may see the hidden or cloaked files but is not able to open it or manipulate its content. Thus, file access is prohibited.

In a third embodiment of the invention, the user may see the file as listed in the directory, and have restricted access to the file contents. The file access may be restricted by the type of application requesting a file access as well as by the procedure within the requesting application. For example, a word processing application may have access to the file for all purposes except deleting the file, and other applications may be denied access altogether.

The invention is intimately connected to the file structure of the computer. All information in most computer system environments are organized into files. Those files are accessed, deleted, printed, copied, moved and, above all, opened, through calls to the file system generated through the operating system by applications and users. All access to files using the key board takes place through the file system. The file system, whether it is the one contained in Windows or Unix operating environment, responds to requests from the user through a series of file calls. These file calls are commands designed to manage files and they can be viewed from a variety of perspectives. A single "open" call made by the user at the next logical layer down consists of perhaps ten individual calls. At the next layer down

these calls disaggregate into many more and at the next layer down break up into a series of hexadecimal calls which the machine translates directly into physical operations.

The virtual cloaker driver operating in accordance with the principles of the invention is a virtual filter driver that loads into memory when the computer boots up. It is 'located' in such a place that it can observe all file calls. The driver can be configured to detect and respond to different calls in different ways. In one embodiment the cloaker driver detects all calls intended to "see" the file in the directory structure, to open it or to manipulate it in any way. When it detects these file specific calls, it essentially cancels them. Thus, when the user opens the computer's directory, she will not be able to see the file name listed. She is also unable to manipulate such files.

In another embodiment, the cloaker driver can provide access to the file for specific processes or programs. Thus, a file invisible to every other program might be readable in Acrobat or Word. In other embodiments, capabilities are added that permit only certain actions or manipulations to take place.

The uses of the cloaker driver are several:

Computers that have multiple users can create password protected file systems that are invisible to other users but visible to the authorized user, and can do this without creating complex disk partitions.

Encryption packages can include invisibility options, increasing the security of the information by maintaining absolute secrecy as to the information's existence.

Users not wanting the time and trouble of encryption could hide files from users, but configure the settings

to allow them to retrieve files whose name they remembered or to display files only after entering the correct password.

A technical advantage of the present invention is the interception of file system calls such that supplemental file management processes can be performed in a manner transparent not only to the user but also to the operating system.

Another technical advantage of the present invention is that supplemental file management is performed in real-time on an ongoing basis transparently to the user of the system.

Additional technical advantages should be readily apparent from the drawings, description, and claims.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete understanding of the present invention and advantages thereof may be acquired by referring to the following description taken in conjunction with the accompanying drawings in which like reference numbers indicate like features and wherein:

FIGURE 1 is a diagram of the cloaker driver inserted to intercept calls from the file system in accordance with the principles of the invention;

FIGURE 2 is a flowchart of an embodiment of the invention in which call processing is performed on a selected type of call coming down the chain from the file system manager and returns coming up the chain to the file system manager;

FIGURE 3 is a diagram of the file system logical layers of the WINDOWS 95 operating system;

FIGURE 4 is a diagram of a typical file system request chain within the file system logical layers of the WINDOWS 95 operating system; and

FIGURES 5A-5G are flowcharts of the processing call and return routines of FIGURE 2.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIGURE 1 is a block diagram of one embodiment of a system for hiding or cloaking files by monitoring file system calls. As shown, the system includes an operating system (OS) layer 10 and an application layer 12. In one implementation, operating system layer 10 is WINDOWS 95 although Windows 98, NT, a Unix based system or indeed any operating system which allows entry points into the file system control may also utilize the same principles of the various embodiments of the invention. Operating system 10 includes a file system 14 (e.g., an installable file system, IFS) that handles calls from applications in application layer 12. In general, device drivers become a device in operating system 10, and applications in application layer 12 use devices to perform tasks through a defined application program interface (API). In accordance with the principles of the invention, high level cloaker application 16 is associated with a cloaker driver 18. For example, cloaker application 16 and cloaker driver 18 can be components delivered by one software vendor and can cooperate to perform functions, such as a hiding or cloaking files as described below.

Cloaker driver 18 operates to monitor and react to calls passing through file system 14. In addition to cloaker application 16, other high level applications 19 can be present in application layer 12 and can send calls through file system 14. Such calls may originate, for

example, from a word processing or other applications or from within the Windows file manager environment. File system calls of interest here are associated with commands "open", "find first", "find next", "delete", and "rename". For ease of description, these commands are also sometimes simply referred to as calls. Generally, operations of interest include any operations which are fundamental in opening, finding remaining or deleting a file. Thus, by way of example, but not by way of limitation, "read", "write", and "get attributes" calls are subsumed for purposes of the embodiment of the invention within the "open" call, since the "open" call will be generated with each of these calls. In a similar manner, for purposes of illustration and not by way of limitation, the "directory" call is subsumed within the "find first" and "find next" calls. Each media device 20 (e.g., hard disk drive, ZIP drive, floppy drive, tape drive, writeable CD ROM drive or other fixed or removable storage media) generally has associated with it a low level driver 22 that handles the interface with media device 20.

In operation, cloaker or monitor driver 18 obtains and analyzes each call passing through file system 14. Cloaker driver 18 can gain access to file system calls, for example, by plugging into file system 14 as a device driver without actually being associated with a particular device. Using this access, cloaker driver 18 can assess whether the call originated from cloaker application 16 or other applications 19 and whether the call is a call of interest, e.g., an "open", "find first", "find next", "delete", or "rename". Cloaker driver 18 can also decide whether to pass the call on to driver 22 associated with media device 20 or to reject or abort the call or to change the return variable or

generate a return variable without passing the call down to subsequent layers. By such operations, the cloaker driver 18 is able to hide files from the user in such a manner that they are completely invisible and will not be found from any application or Windows system command.

The cloaker application 16 is implemented to permit the user to cloak or hide files. Running the cloaker application permits the user to select desired files for cloaking and then to apply the cloaking operation to such files which is done by passing the selected file names to the cloaker driver 18 so that they may be added to a lookup table or database within the cloaker driver 18. Preferably, the user is asked to select a password prior to initiating the cloaking process and only the use of this password permits the user to de-cloak the previously cloaked files. Most preferably, the password would be different from the Windows user password. In this manner two or more users could use the same computer hardware but have access to, and indeed even know of the existence of, different subsets of the stored files. User1 could keep confidential files hidden on the system from User2 and visa versa. Not only are the contents of the files hidden, but the very existence of the files are hidden since, the cloaked files, in one embodiment of the invention, will not show up in any directory listing nor can they be accessed in any manner by the operating system until they are de-cloaked by the user who must first enter the appropriate password.

As an added safety precaution, the files cloaked can be encrypted automatically after (or before) the file names are added to the cloaker driver's lookup table or database. After entering of the correct password, the encrypted file is decrypted upon file opening and encrypted again upon file closing. The cloaked file will

remain encrypted when closed until it is no longer desired to be cloaked. When the authorized user removes the cloaking using the cloaker application 16, as performed by removing the file name from the cloaker driver lookup table or database, the file will be automatically decrypted. There are many ways in which the cloaker application and cloaker driver may affect the encryption and decryption processes and the above description is merely one implementation. The basic usefulness of the encryption results from someone booting the system from a floppy disc so as to run the system on a different operating system which would mean that the cloaker driver program would not be executing. Encrypting the files would thus prevent unauthorized access to the file by rendering their content difficult if not impossible to decipher in the floppy boot situation.

FIGURE 2 is a flowchart of an embodiment of a method according to the present invention for monitoring file system calls to a file system structure of an operating system and for controlling the original call or returned variables. The method of FIGURE 2 is implemented by the virtual cloaker software driver 18 with the purpose of hiding or cloaking files that may be stored on media device 20. In one embodiment, the method of FIGURE 2 can be implemented using a vendor supplied driver (VSD) executing within the installable file system (IFS) of WINDOWS 95.

As shown in FIGURE 2, in step 110, a file system call is intercepted. In general, the file system call is intended to perform some function with respect to data stored on a media device 20 but is intercepted before being able to complete that function. In step 112 of FIGURE 2, it is determined whether the type of call

intercepted is one that should be processed. Generally, the calls of interest would consist of any calls that could find or identify a file which the user or a previous user cloaked. By cloaking, a file is rendered invisible from both the user's perspective and the operating system's perspective and can't be accessed, changed, modified, deleted, listed or even found by the operating system. Thus, once a file is cloaked, calls of interest would include "open", "find first", "find next", "delete", and "rename". Generally, if a file is cloaked, the call attempting to perform some action with reference to the cloaked file or its returned variable is controlled so as to hide or cloak the file. Such control may include generating a return variable such as an error code or other actions as explained below in relation to FIGURE 5. If such a call is not identified in step 112, then in step 114, the original call is passed on through the file system. If the call should be processed, i.e., it is one of the calls of interest, then in step 116, some type of call processing is performed to ensure that the cloaked file is not made known to the user. The particular type of processing is dependent upon the type of call and is explained more fully in relation to FIGURES 5A-5G.

According to the present invention, the call processing of step 116 can be accomplished transparently both to the user and to the calling system application. After the call processing, the call is returned, in step 118, to the calling system application, as for example, application 19 in FIGURE 1.

FIGURE 3 is a diagram of file system logical layers of the WINDOWS 95 installable file system (IFS) 14. The installable file system is described, for example, in "Examining the Windows 95 Layered File System: Adding

Functionality to Block Devices," Mark Russinovich and Bryce Cogswell, 1997 [http://www.ddj.com/ddj/1995/1995.12/russinov.htm as of September 4, 1997]. As shown in FIGURE 3, the installable file system is made up of thirty two logical layers, each containing one or more virtual devices through which block-device requests pass. For typical hardware, most of the logical layers are empty. For example, for hard disk drives, a file system request (or call) will usually only pass through about five virtual devices on the way to the hardware.

FIGURE 4 is a diagram of a typical file system request chain or path within the file system logical layers of the WINDOWS 95 operating system. As shown in FIGURE 4, a typical path begins at the IFS manager 122 and moves to the file system driver 124. The request then moves to a type specific driver 126 and, in this case, to the vendor supplied cloaker driver 18. After vendor supplied cloaker driver 18, the request falls to a port driver 22 and to the media drive 20 (e.g., hard drive or other storage device). After the request is completed at the physical level, the request returns up the chain to the calling system application.

In FIGURE 3, the numbers on the lefthand side represent the layers of abstraction with the smallest numbers representing higher layers of abstraction. The topmost layer is the entry point to the file system. Higher numbers are closer to the hardware, and the highest number (bottom layer) represents the virtual devices that access the hardware directly. An input/output (I/O) supervisor (IOS) manages requests as they pass through the file system hierarchy. Each virtual device on the chain can select requests based on the logical or physical drive to which the request is

directed. The devices can also view the result of a request as it passes back up the chain to the application. Furthermore, the virtual device drivers (VxDs) on the chain can service requests themselves and not pass them to lower levels, or they can generate requests themselves.

Processes can occur at each level of the installable file system, but most block devices do not require an entry at each level in the chain. At the top of the logical layer structure, the IFS manager layer manages high-level I/O requests from applications. It takes a call directed at a specific logical drive and passes it down the correct call-down chain to the appropriate volume tracker, file system driver (FSD), and so on. Volume trackers work with groups of devices with identical removability rules. For example, a CD-ROM volume tracker ensures that a CD with a file system on it is in the drive before it will allow any requests to pass through to lower layers. File system drivers (FSDs) work with all devices of a particular type, such as hard disks or CD-ROM devices. They take incoming logical requests generated by the IFS manager and translate them into physical requests to pass to lower levels. In addition, FSDs can initiate logical error recovery for devices such as disks.

Type specific drivers (TSDs) work with all devices of a particular type. They take a logical request generated by an FSD and translate it into a physical sector request. They generally reside in the same layer as their corresponding FSDs, but are lower in the chain. SCSI-izers are next in the chain and are used because SCSI devices require more complex request packets than other devices such as the more prevalent IDE/ESDI devices. SCSI-izers take a general physical request and

create a SCSI Request Block (SRB) that contains detailed, SCSI-specific information about the request such as the Logical Unit Number (LUN) and Target (SCSI targets can have up to seven LUNs hanging off them).

Vendor supplied drivers (VSDs) provide a special layer for third-party developers under WINDOWS 95. Consequently, the VSD layer functionality is determined by the VSD writer. Conventional uses include: block-device monitors, low-level secondary disk caches (caching in flash memory, for example), data encryption, and RAID disk management.

SCSI port drivers take incoming requests and determine which SCSI miniport driver should field them. Multiple SCSI types can be loaded onto the same system, each of which may require a custom SCSI miniport driver. The SCSI port driver is also in charge of initializing the miniport drivers. SCSI miniport drivers (MPDs) are the hardware drivers for SCSI devices. They manage the interrupt and I/O port-level interaction with the device to carry out requests from above. They can also perform adapter-specific error recovery.

Port drivers (PDRs) (for non-SCSI hardware) carry out analogous functions as the SCSI port and miniport drivers. They provide 32-bit disk access interacting directly with the hardware to perform I/O. The real mode mapper (RMM) is used in certain situations where WINDOWS 95 can not provide a port drive. With the introduction of plug-and-play BIOS, and by including many hardware specific port drivers, WINDOWS 95 can usually provide 32-bit access for most disk hardware. However, Windows 95 might be run on an older personal computer with esoteric hardware, so it must make allowances for the case where it can not provide a port driver to handle disk I/O in protected mode. A system might also use real-mode disk

driver software that provides functionality not available in the WINDOWS 95 protected mode counterpart. For these situations, the last entry on the chain of protected mode virtual device drivers is an RMM instead of a port driver. RMMs call down to a real mode driver to perform hardware I/O and return results up the file system chain. Real mode drivers are hardware drivers required by the hardware or software configuration of a particular system. However, use of real mode drivers is discouraged because performance can suffer (due to the overhead of transitions from protected to real mode and slower execution in real mode), but makes allowances for them for flexibility and backward compatibility.

In general, the upper layers of the file system structure are written by MICROSOFT as part of WINDOWS 95, while the lower layers are provided by disk drive manufacturers. Consequently, the layer typically used for development by third-party developers is the virtual supplies driver (VSD) layer. As mentioned above, according to the present invention, a VSD can be used to intercept file system calls and perform call processing so as to ensure that any cloaked file is hidden from the user.

In one implementation of the present invention, a vendor supplied driver is used to intercept file system "open", "find first", "find next", "delete", and "rename" calls to as to hide cloaked files from the user. Interception of the these calls occurs in step 110 of FIGURE 2. The above calls are then identified, in step 112, as ones for which call processing will be performed per step 116.

FIGURES 5A-5G are flowcharts of one embodiment of call interception and processing applicable to processing the calls and returns of the "open", "find first", "find

next", "delete", and "rename" calls. File system call interception is done in step 210. After call interception, the call is evaluated to see if it is the type of call of interest. This evaluation is performed in steps 212 through 220. In step 212, the cloaker driver determines if the call is an "open" call; in step 214 if the call is a "find first" call; in step 216 if the call is a "find next" call; in step 218 if the call is a "delete" call; and in step 220 if the call is a "rename" call. Any "no" determination moves down the flowchart 212-220. If none of the calls are of a type of interest, the call is forwarded to the next layer in step 222A, and the call is thus passed on without modification by the cloaker driver 18.

If in step 212, the call is determined to be the "open" call, the file name is examined to determine if the file is cloaked. The cloaker driver 18 looks up the file name in a lookup table or database to see if the file has been previously cloaked. If the file is not cloaked, the cloaker driver 18 sends the call to the next layer in step 222A. If the file is cloaked, the cloaker driver 18 generates return variable in step 232 which passes to the file system manager and ultimately indicates to the user that the file is not found or some error has taken place. For example, the cloaker driver 18 can sets the error code to a value of 2 indicating a "file not found" message and sets the return variable to "-1" to indicate a failure of the call execution. As an added safeguard, the cloaker driver 18 sets the file handle to NULL which indicates to the IFS manager that the handle does not exist. The net result of step 232 is not only to prevent the file from opening, but to indicate to the user that the file itself is not existent in the system. Thus, assuming the user had a suspicion

that a certain file, say "SECRET.doc", existed in the system, any attempt to access the file using the "open" command is defeated by returning the null handle and error messages per step 232.

Step 214 examines the "find first" call. If this call is found, it is passed down the system layers and upon return, the retrieved file name is examined in step 242 to see if the request has targeted a cloaked file. If it is not cloaked, the driver proceeds to step 222B and passes the call directly up the chain to the IFS manager. If in step 242 the file is a cloaked file, the cloaker driver 18 proceeds to step 244 and executes a "find next" function. Upon return in step 246, the return variables are examined to see again if the targeted file is cloaked. If it is not cloaked, the program proceeds to step 222B as before. If the targeted file is cloaked, the program loops back to step 244 to again execute another "find next' function. In this manner, the name and even the existence of the cloaked file is kept hidden from the user, and only the first found non-cloaked file will be displayed to the user.

A similar procedure as outlined above is performed for the "find next" call. Thus, if the "find next" call is identified in step 216, the call is passed down the chain in step 250 to return the file name, and if the name examined in step 252 corresponds to a cloaked file, another "find next" call is passed down the chain in step 254 without permitting the return of the first file name to the IFS manager. The return of this "find next" function is examined in step 256, and if it does not point to a cloaked file, the program proceeds to step 222B to pass the return to the IFS manager. If the return identifies a cloaked file in step 256, the program loops again to step 254 to generate yet another "find

next" function. In this manner, the find next function never identifies to the OS the identity of a cloaked file.

The "find first" and "find next" calls are typically used to list directories and thus the procedures outlined in the flowcharts effectively hides any cloaked files by preventing them from ever being displayed on a directory listing.

Step 218 examines the "delete" call. If this call is found, a determination is made in step 260 as to whether or not the targeted file is cloaked. If the file is not cloaked, step 222A is performed to pass the call down to the next layer. If the targeted file is cloaked, an error message "file not found" is displayed by setting the error code to 2, by setting the return code to -1 indicate a failure and returning control back to the file system manager. In this manner, the user is not able to delete the targeted file and indeed, the system acts as if the file doesn't even exist - exactly the desired outcome to cloak a file.

In step 220, the "rename" call is examined. If this call is found, the cloaker driver 18 proceeds to step 270 to determine if the targeted file is cloaked. If the file is not cloaked, the call is passed down to the next layer in step 222A. If the file is cloaked, a return variable is generated in step 272 to set the error code to 2 and to return -1 as done in set 262.

Is should be understood that the cloaker operations are performed in real-time whenever the user initiates a procedure that results in one of the "open", "find first", "find next", "delete", and "rename" calls being generated.

The monitoring of the file calls and returns as indicated above is performed by the cloaker driver 18.

Generally, the files and returns are part of an "IFS request packet" with the request traveling down from the IFS manager to the lower level drivers and the return or return variables inserted into the packet and sent up the chain toward the IFS manager from the lower level drivers.

Extending the principles illustrated in FIGURES 5A-5G to the Windows NT environment is straighforward for the "open", "delete", and "rename" calls. For "find first" and "find next" however, more processing is required in order to manipulate the index buffers which are returned describing the list of files in the directories. The NTFS file system acts in a transactional database manner and indexes its files for quick look-up and display purposes. Such indexes are prefabricated. Thus, to cloak files, the files must be removed from the memory segments once the index runs are returned to the cloaker driver.

The embodiment of the invention shown in FIGURES 5A-5G essentially prevents the user of the operating system from learning of the very existence of any files which are cloaked. However, in some applications, it may be desirable to permit the user to list the cloaked file in the directory but to otherwise not be able to access the files. The file is said to be access-prohibited. In essence, the user is denied effective access to the file in that the user can only learn of existence of the file by viewing the file name in a directory listing. In this embodiment, the directory listing is not password protected, but access beyond viewing the file is password protected so as to permit only the person with access authorization to access the files. The cloaked file in this embodiment is preferably encrypted to ensure that the file content remains unavailable. Such an embodiment

has application in permitting anyone to screen computers or memory devices searching for sensitive files without fearing that the sensitive files will be accessible to the screening operator.

The implementation of the above embodiment of the invention merely requires one to remove the find open and find next calls in steps 214 and 216 respectively. In this manner directory listing will be possible but open, delete and rename processes will not be permitted.

Yet another embodiment of the invention utilizes permissions or restrictions applicable to a given process or application and to selected procedures within the application. The file is said to be use-restricted. For example, one may utilize this embodiment of the invention to permit a word processing program to read a file and otherwise edit and copy a file but not to delete a file. In this embodiment, the monitoring system of FIGURES 5A-5G would be modified to remove all call monitoring except for the delete call monitoring in steps 218, 260, 222A and 262. An added logical inquiry would determine if the monitored call or request packet contained a designated word processing application such as Word. If the cloaker driver determined that the requesting application was Word, and that a delete call was send from the file system manager, then the sequence would branch to the delete monitoring steps 218, 260, 222A and 262. In this manner, the embodiment of the invention does not cloak a file but rather restricts use of the file at least as to one feature the would normally be available in a given application.

The cloaker application 16 permits the user to select (de-select) which files are to be stored in (removed from) the database or lookup table of the cloaker driver 18. The cloaker application 16 also

presents a user interface for insertion of user passwords, encryption options an other types of permissions applicable in the cloaked, access-prohibited and use-restricted modes of operation. The cloaker application initially list all the files in a selected directory and permits the user to highlight those files which are desired to be cloaked. The highlighting effectively selects the files to be cloaked. By entering the selection (pressing enter or an "o.k." command in a dialog box) the user causes the selected files to be stored in the cloaker driver database or lookup table. The user preferably is required to enter a password so that the user will be able to de-cloak the file at a later time or add new files to be cloaked. The password may be entered initially but may alternatively be entered after file selection, but at any rate is entered before the execution of the driver program which monitors the calls from the file system manager. To de-cloak files, the user re-enters her password which is recognized by the cloaker application which then extracts the file names listed in the cloaker driver database (or lookup table) so they may be displayed preferably during a normal list directory command. Files which are cloaked are distinguished in the directory listing as by highlighting or other means so the user will know which files to de-select thereby removing them from the database or lookup table and thus de-cloaking them. At the same time other files may be selected for cloaking, access prohibition and/or use-restriction.

The modes of cloaking, access prohibition and use-restriction may be selected for each file using dialog boxes or other inputs from the GUI of the cloaker application. The program flow executes each mode on a per file basis. Thus files A and B may be cloaked, file

C may be access prohibited and files D and E use-restricted. Initial selections of the mode/file combinations are made by the user during a file/mode selection process using the application program.

In yet another mode of operation, the files to be hidden need not be selected by the user of the operating system, but could be pre-selected by a third party vendor selling some application for installation on the operating system. The third party vendor may want to keep certain data or .exe files from being accidentally deleted by the user of the operating system and yet have these files available for access by the application. For example application X may contain files Y and Z which the vendor desires to be cloaked, i.e., rendered invisible to the operating system, except if application X desires access. In this case application X contains a cloaker driver which cloakes files Y and Z from the operating system except as to application X. This operation is essentially a use-restriction of the files Y and Z. If application X tries to access files Y and Z, the operating system is able to provide full access and use permissions through the file manager. If the user tries to gain access of any kind outside of application X, such actions are negated and the operating system acts as if the files do not exist (i.e., they can not be found)..One application other than preventing accidental deletion cf a file by the user is to prohibit file copying by the user. Such application is important in many applications such as the internet distribution of audio and video content files. In this case application A can "play" the cloaked file because application A has use permissions. The user, however, is not able to otherwise manipulate the file (move, copy, rename, delete, list directory) through the file system manager of the operating system.

Of course, the cloaker driver may be utilized to give the user certain use permissions other than through application A, as for example the list directory and delete commands but prohibit other commands to prevent duplication or retransmission of the file over the internet.

The cloaker drive of the various embodiments of the invention may be downloaded to the user via the internet or provided to the user via CD. In the download situation, the program is stored on a server memory and downloaded in the normal course of e-commerce.

As explained above, cloaking or hiding a file prevents the file from being listed on a directory listing and otherwise prevents the operating system of a computing device from accessing the file. Thus, the file may not be opened, deleted, renamed or otherwise accessed from the file system of the computer operating system. The cloaker driver identifies calls from and returns to the file manager wherein at least of the calls and returns is associated with a specific file identification data such as the file name. This association may be a direct reference to the name in the call or return packet or an indirect reverence to pointers which in turn identify the file name or identification data so as to uniquely identify the file. In this manner, the operations to be negated (open, find first, find next, delete, and rename) may be uniquely associated with a specific file whose identification may be checked against the cloaker driver database or lookup table so as to apply the negation only to those files which are selected to be cloaked, access-prohibited or use-restricted.

It should be understood that the invention has applicability to any computing device including appliances where a file system is controlled by an

operating system. Appliances running the Microsoft CE operating system may also make use of the invention as for example in set-top boxes and a multitude of small appliances including personal data assistants. Such devices all have operating systems controlling file storage and retrieval and the cloaker driver may effectively be used to cloak such files or provide access prohibition or use restrictions or any combination of these modes on a file by file basis just as in the case of a PC or laptop computer. Indeed, in many circumstances, it is envisioned that a PC or laptop computer will be able to connect up to such appliances so file cloaking, access prohibition and use restrictions are desirable and useful enhancements to the system operation. Thus, the term "computing device" as utilized in the appended claims is intended to cover all such data processing devices, PDA's, appliances and indeed any computing apparatus that has a file system and an operating system.

It will be appreciated that the primary purpose of the cloaker application program is to act as a GUI interface to the user for input of the passwords, encryption options, file/mode selections as explained above. The primary purpose of the cloaker driver is to monitor and process calls from and returns to the file system manager. A device I/O controller interfaces between the application program and the driver. In the context of the appended claims, the application and driver functions are recited in terms of the cloaker driver or virtual driver, but one can readily appreciate that such terminology is to be understood by one of skill in the art in the context of the above distinctions between the application software and the driver software.

Although the present invention has been described in detail, it should be understood that various changes, substitutions and alterations can be made hereto without departing from the spirit and scope of the invention as defined by the appended claims.

WHAT IS CLAIMED IS:

1. A method for hiding files accessed through a file manager of an operating system of a computing device, comprising the steps of:

a) selecting at least one file to be hidden,

b) storing identification data of the at least one selected file,

c) identifying at least one of (1) calls from the file manager and (2) returns to the file manager which is associated with said stored identification data, and

d) controlling at least one of said identified calls and returns to conceal the existence of the selected file thereby hiding the selected file.

2. The method of claim 1 wherein the step of identifying includes identifying calls and returns associated with commands selected from the group consisting of "open", "find first", "find next", "delete", and "rename" and wherein controlling said calls and returns has the effect of negating said commands.

3. The method of claim 1 further comprising the step of encrypting the selected file.

4. The method of claim 1 further comprising the step of:

entering a user password as a necessary step prior to the monitoring step.

5. The method of claim 1 further comprising the steps of:

entering a user password,

if the password matches a pre-stored password, displaying a file name of said at least one selected file,

using said displayed file name, deselecting said at least one selected file to permit the existence of the deselected file to no longer be hidden from the operating system.

6. The method of claim 1 further comprising the steps of:

entering a user password as a necessary step prior to the monitoring step so as to store said user password into said system,

re-entering said user password after said controlling step,

comparing said re-entered user password with said stored user password and if they match, then displaying a file name of said at least one selected file,

using said displayed file name, deselecting said at least one selected file to permit the existence of the deselected file to no longer be hidden from the operating system.

7. The method of claim 6 further comprising the step of encrypting said selected file and decrypting said deselected file.

8. The method of claim 7 wherein the step of decrypting is performed upon opening a file.

9. The method of claim 1 wherein said identification data comprises a file name.

10. A method of prohibiting access to files in a computing device having an operating system and a file system manager comprising the steps of:

a) storing identification data of files desired to be access-prohibited,

b) identifying request packets to and from the file system manager which are associated with said stored identification data, and

c) processing said request so as to prohibit the user of the operating system from accessing said file desired to be access-prohibited thereby rendering said file access-prohibited.

11. The method as recited in claim 10 wherein the processing step comprises prohibiting the user from learning of the existence of the access-prohibited file by any calls to the file system manager corresponding to list a directory, open, find , delete or rename the access-prohibited file.

12. The method as recited in claim 10 wherein the processing step further comprises prohibiting the operating system from performing any user command to list the name of any access-prohibited file in a directory, or to open, find, delete or rename any access-prohibited file.

13. The method of claim 10 further comprising the steps of encrypting said access-prohibited file.

14. A method of restricting use of files in a

computing device having an operating system and a file system manager comprising the steps of:

a) storing identification data of files desired to be use-restricted,

b) using said stored identification data, monitoring request packets to and from the file system manager to identify, request that would, except for step c), permit the user of the operating system to fully use files desired to be use-restricted, and

c) processing said request so as to prohibit the user of the operating system from fully using said file desired to be restricted in use thereby rendering said file use-restricted.

15. A method of cloaking files in a computing device having an operating system and a file system manager comprising the steps of:

a) storing identification data of files desired to be cloaked,

b) identifying request packets to and from the file system manager which are associated with said stored identification data, and

c) processing said request so as to cloak files associated with said stored identification data.

16. A method of cloaking files in a computing device having an operating system and a file system manager comprising the steps of:

a) storing identification data of files desired to be cloaked,

b) identifying request packets to and from the file system manager which are associated with said stored identification data, and

c) processing said request so as to negate file commands of open, find first, find next, delete and rename.

17. A computer system operable for cloaking selected files comprising:

a. a processor including an operating system,

b. a memory device for storing files including cloaked files,

c. a file system manager, governed by said operating system, for managing files in said memory device,

d. a virtual cloaker driver operable for:

(i) monitoring at least one of (1) request packets from said file system manager and (2) request packets to said file system manager,

(ii) identifying monitored request packets associated with a pre-selected list of files desired to be cloaked, said files in said pre-selected list stored in said memory and defining said cloaked files,

(iii) processing said request so as to prohibit the user of the operating system from accessing said cloaked files and from listing said cloaked files in a file directory, thereby rendering the file invisible to the user and the operating system.

18. The system of claim 17 wherein said virtual cloaker driver processes is further operative to respond to a user password for permitting modification of said pre-selected list to thereby permit addition and removal of files from being cloaked.

19. The apparatus of claim 17 further wherein said cloaker driver encrypts said cloaked files stored in said

memory.

20. A computer system operable for restricting use of selected files comprising:

a. a processor including an operating system,

b. a memory device for storing files including said use-restricted files

c. a file system manager, governed by said operating system, for managing files in said memory device,

d. a virtual driver operable for:

(i) monitoring at least one of (1) request packets from said file system manager and (2) request packets to said file system manager,

(ii) identifying monitored request packets corresponding to a pre-selected list of files to which a use restrictions is desired, said files in said pre-selected list stored in said memory and defining said use-restricted files,

(iii) processing said request so as to restrict the user of the operating system from full using files on said pre-selected list thereby rendering same use-restricted.

21. A method of restricting the use of a file comprising the steps of:

a) selecting at least one file to be restricted,

b) storing identification data of the at least one selected file,

c) using the stored identification data, monitoring at least one of (1) calls from the file system and (2) returns to the file system to identify at least one of said monitored calls and returns that, except for step

d), would permit unrestricted use of said at least one selected file, and

d) controlling at least one of said identified calls and returns to restrict the use of said selected file thereby restricting the selected file.

22. A memory storage device storing a computer program which when executed on a computing device having an operating system and a file system manager causes said computing device to cloak selected files by performing the operations of:

a) storing identification data of files desired to be cloaked,

b) identifying request packets to and from the file system manager which are associated with said stored identification data, and

c) processing said request so as to cloak files associated with said stored identification data.

23. A memory storage device storing a computer program which when executed on a computing device having an operating system and a file system manager causes said computing device to restrict use of selected files by performing the operations of:

a) storing identification data of files desired to be cloaked,

b) identifying request packets to and from the file system manager which are associated with said stored identification data, and

c) processing said request so as to restrict the use of files associated with said stored identification data.

24. A memory storage device storing a computer

program which when executed on a computing device having an operating system and a file system manager causes said computing device to prohibit access to selected files by performing the operations of:

a) storing identification data of files desired to be cloaked,

b) identifying request packets to and from the file system manager which are associated with said stored identification data, and

c) processing said request so as to prohibit access of files associated with said stored identification data.

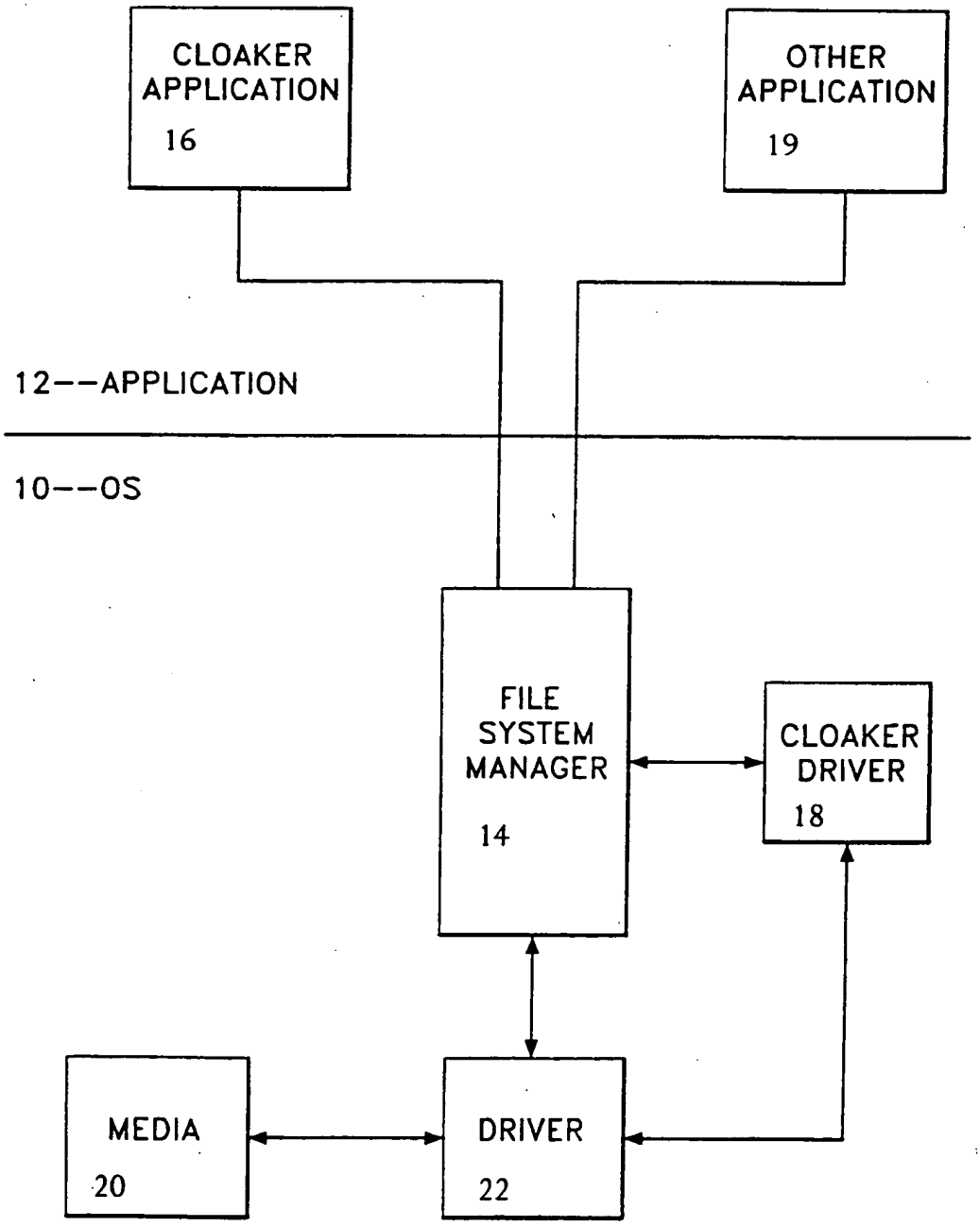

CLOAKER APPLICATION 16
OTHER APPLICATION 19
12--APPLICATION
10--OS
FILE SYSTEM MANAGER 14
CLOAKER DRIVER 18
MEDIA 20
DRIVER 22

FIG. 1

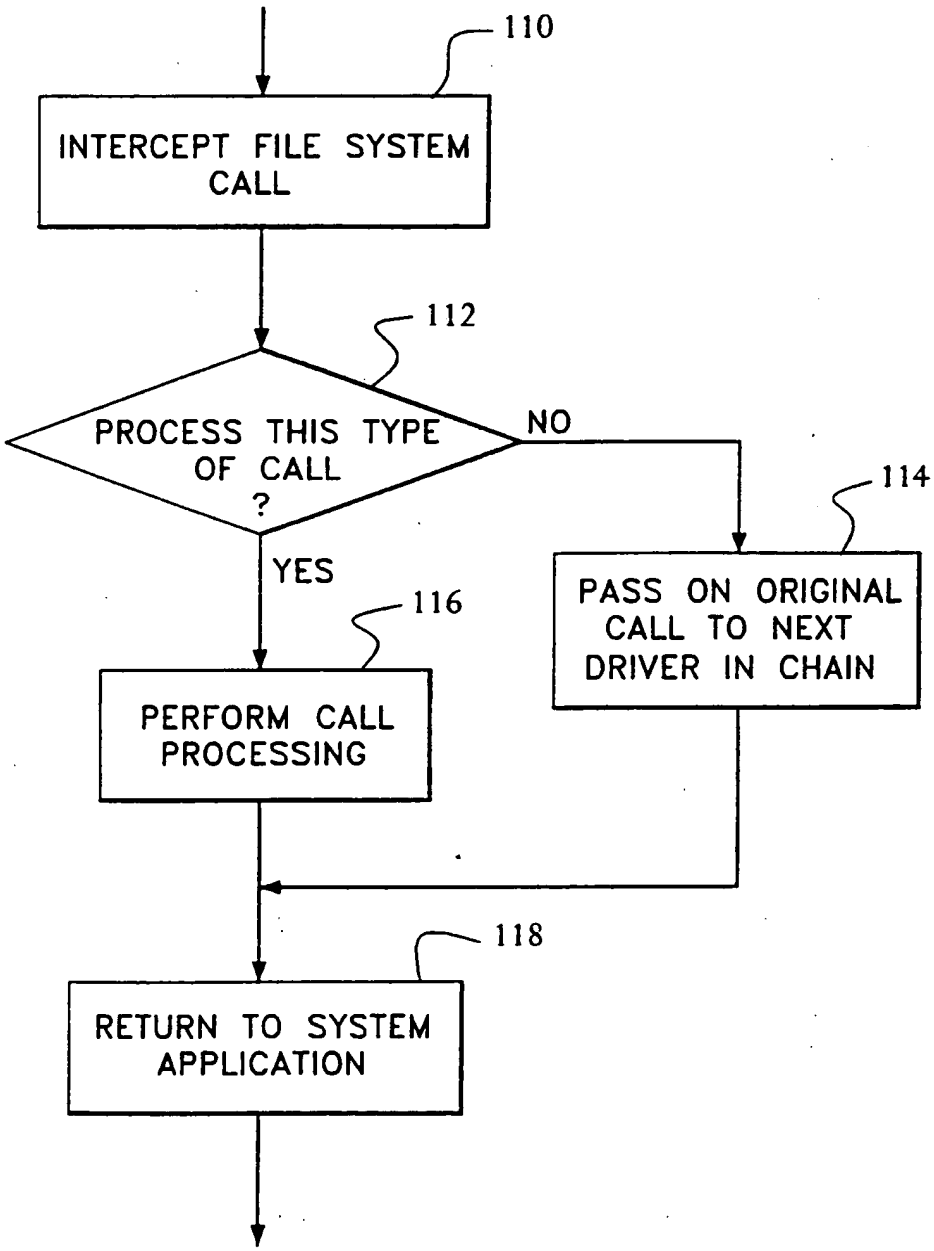

110
INTERCEPT FILE SYSTEM CALL
112
PROCESS THIS TYPE OF CALL ?
NO
YES
114
PASS ON ORIGINAL CALL TO NEXT DRIVER IN CHAIN
116
PERFORM CALL PROCESSING
118
RETURN TO SYSTEM APPLICATION

FIG. 2

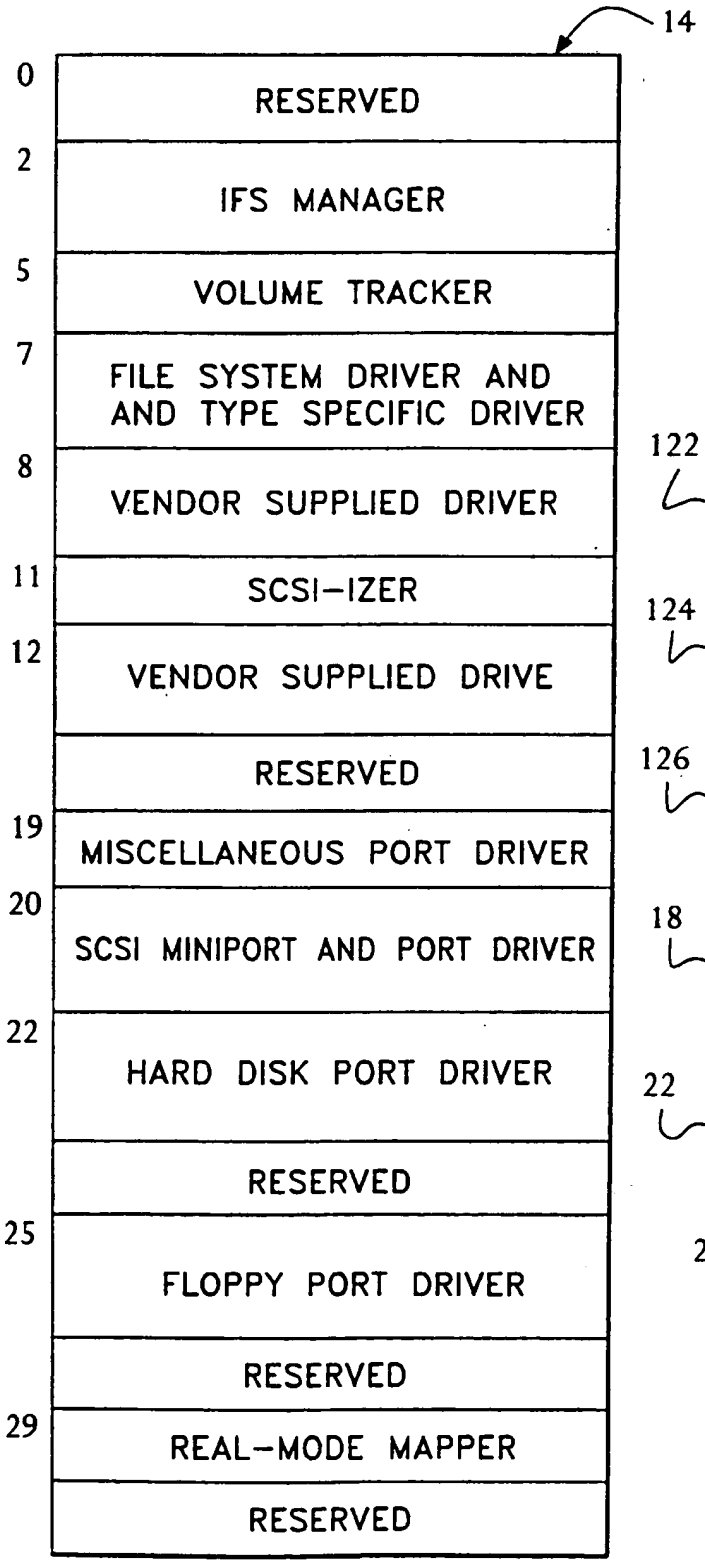

14
0
RESERVED
2
IFS MANAGER
5
VOLUME TRACKER
7
FILE SYSTEM DRIVER AND AND TYPE SPECIFIC DRIVER
8
VENDOR SUPPLIED DRIVER
11
SCSI-IZER
12
VENDOR SUPPLIED DRIVE
RESERVED
19
MISCELLANEOUS PORT DRIVER
20
SCSI MINIPORT AND PORT DRIVER
22
HARD DISK PORT DRIVER
RESERVED
25
FLOPPY PORT DRIVER
RESERVED
29
REAL-MODE MAPPER
RESERVED

FIG. 3

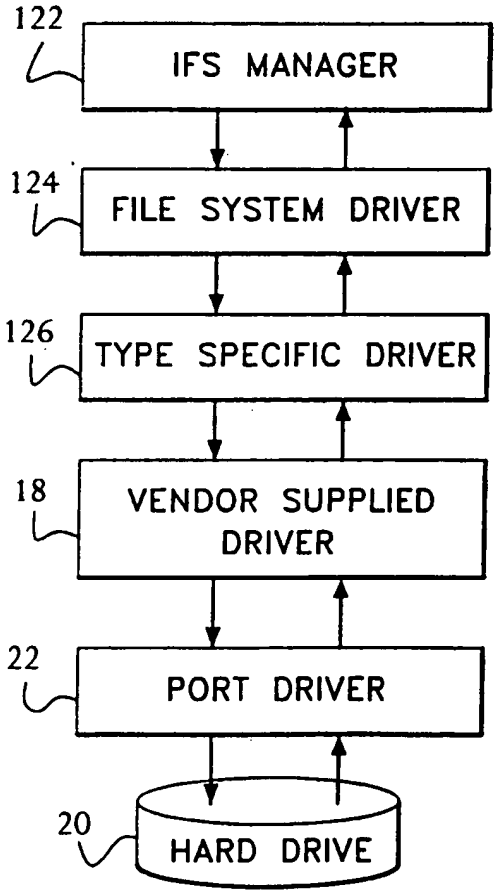

122
IFS MANAGER
124
FILE SYSTEM DRIVER
126
TYPE SPECIFIC DRIVER
18
VENDOR SUPPLIED DRIVER
22
PORT DRIVER
20
HARD DRIVE

FIG. 4

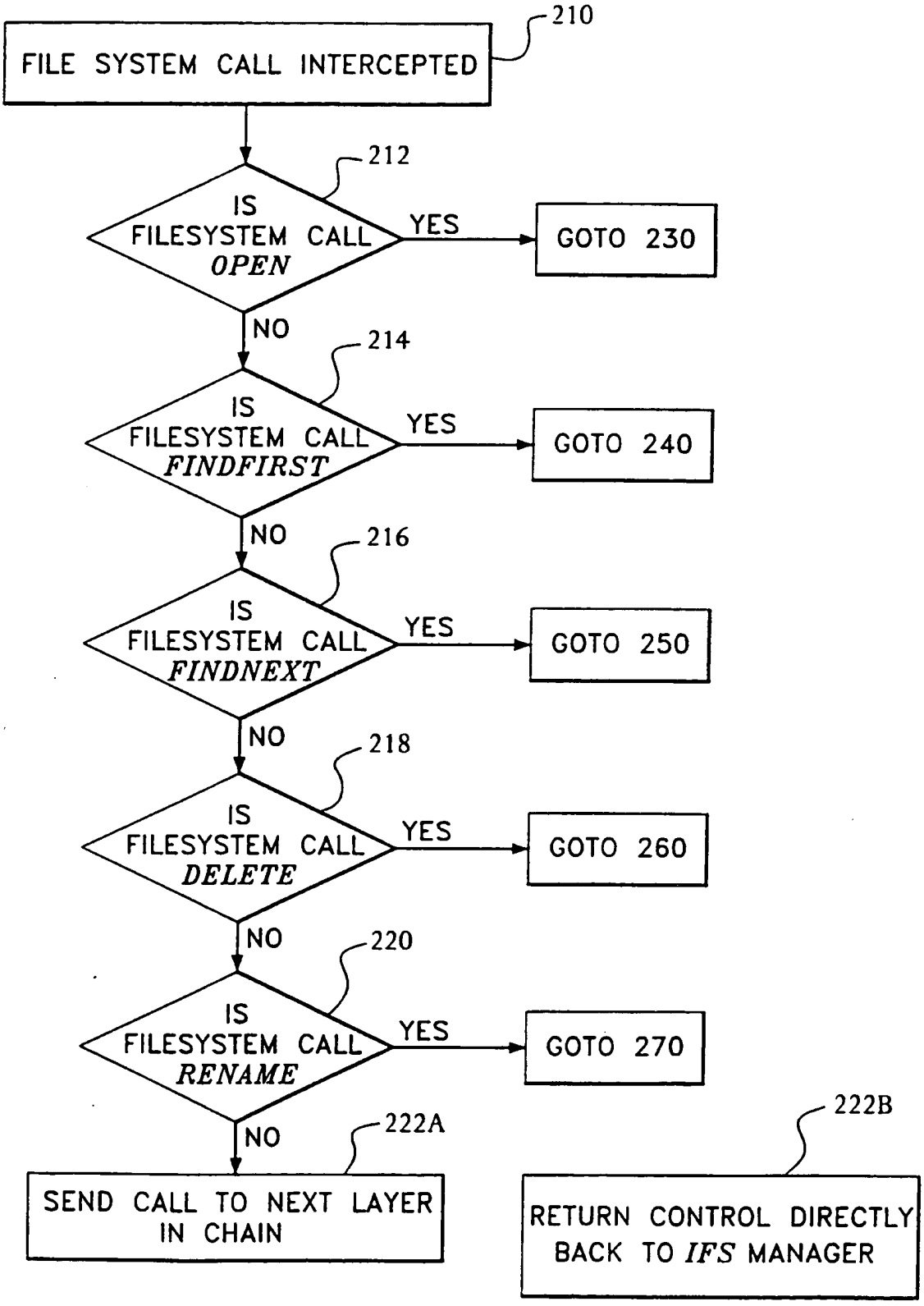

FILE SYSTEM CALL INTERCEPTED
210
IS FILESYSTEM CALL OPEN
212
YES
GOTO 230
NO
IS FILESYSTEM CALL FINDFIRST
214
YES
GOTO 240
NO
IS FILESYSTEM CALL FINDNEXT
216
YES
GOTO 250
NO
IS FILESYSTEM CALL DELETE
218
YES
GOTO 260
NO
IS FILESYSTEM CALL RENAME
220
YES
GOTO 270
NO
SEND CALL TO NEXT LAYER IN CHAIN
222A
RETURN CONTROL DIRECTLY BACK TO IFS MANAGER
222B

FIG. 5A FIG. 5B

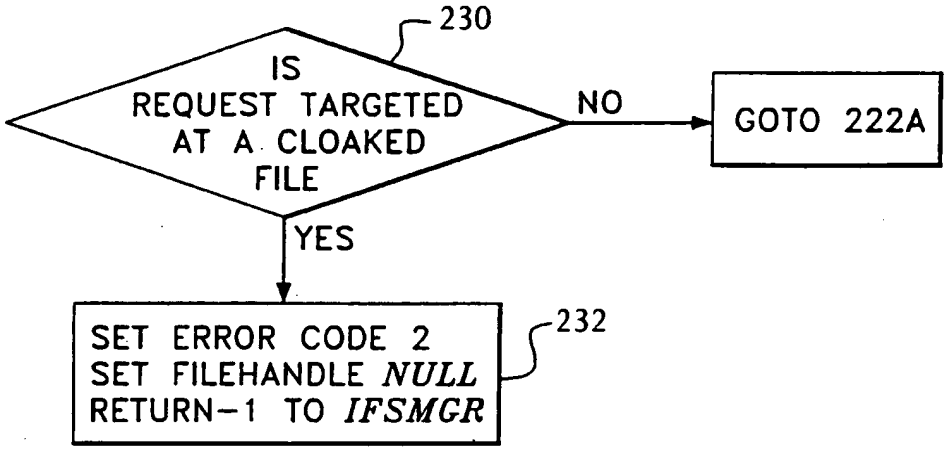

230
IS REQUEST TARGETED AT A CLOAKED FILE
NO
GOTO 222A
YES
SET ERROR CODE 2
SET FILEHANDLE *NULL*
RETURN-1 TO *IFSMGR*
232

FIG. 5C

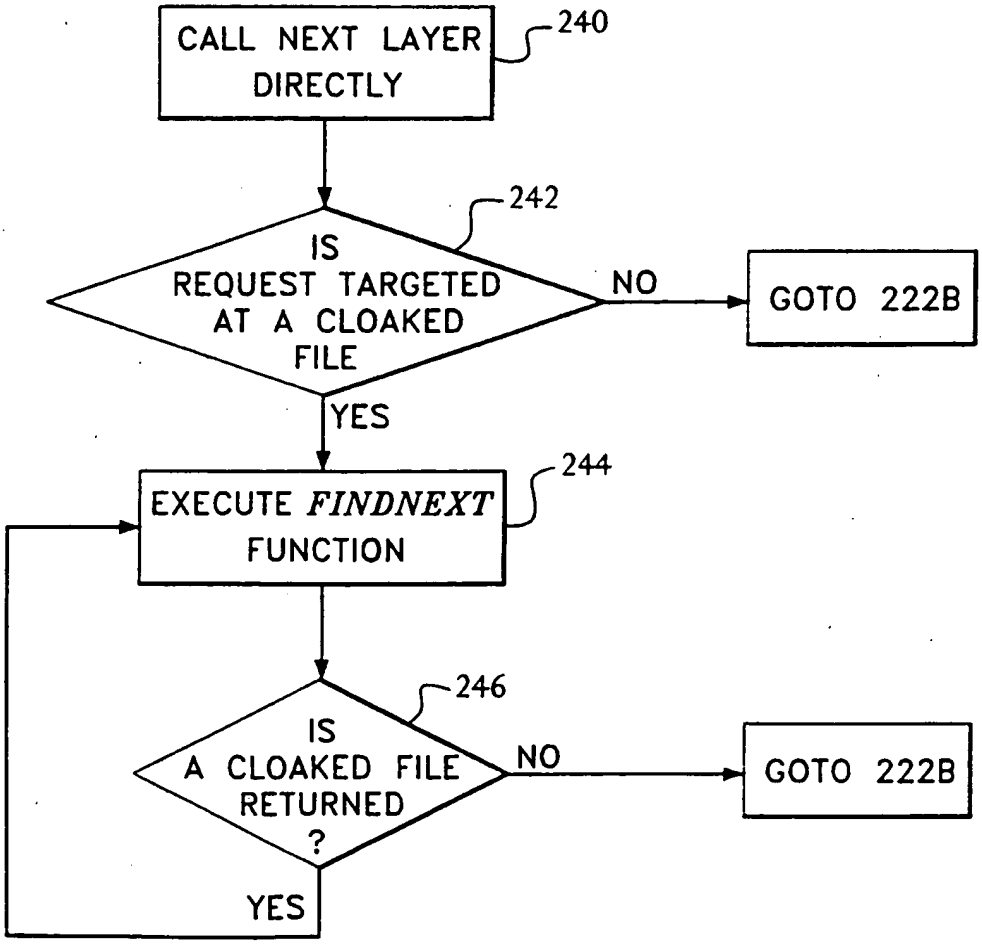

CALL NEXT LAYER DIRECTLY
240
242
IS REQUEST TARGETED AT A CLOAKED FILE
NO
GOTO 222B
YES
244
EXECUTE *FINDNEXT* FUNCTION
246
IS A CLOAKED FILE RETURNED ?
NO
GOTO 222B
YES

FIG. 5D

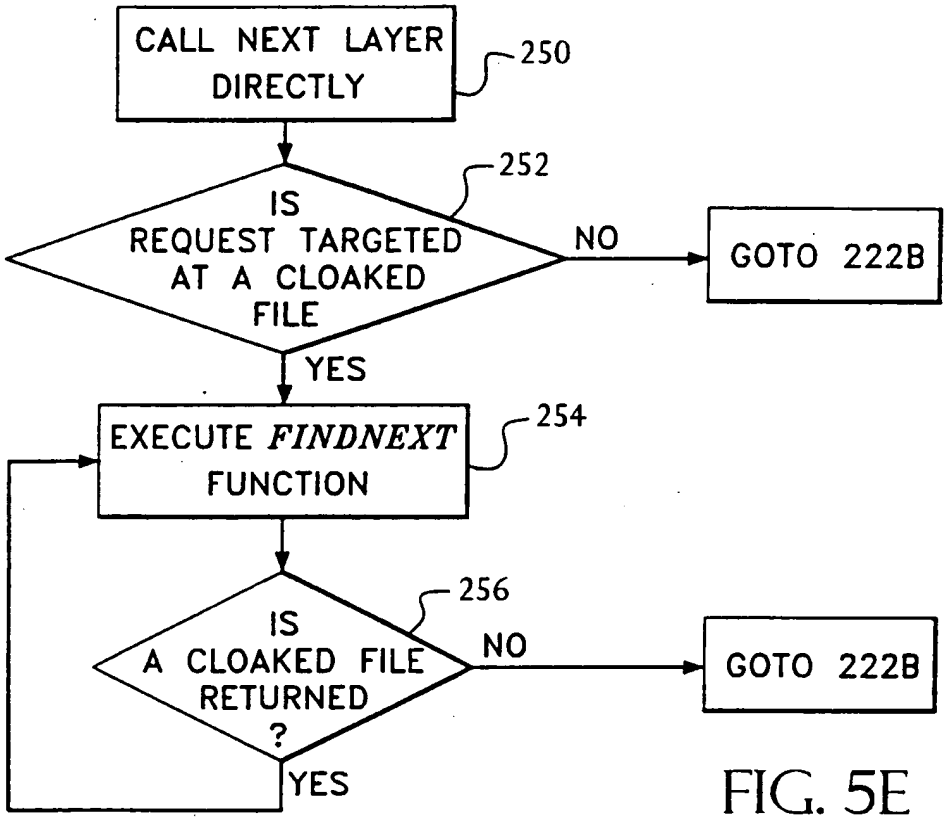


FIG. 5E

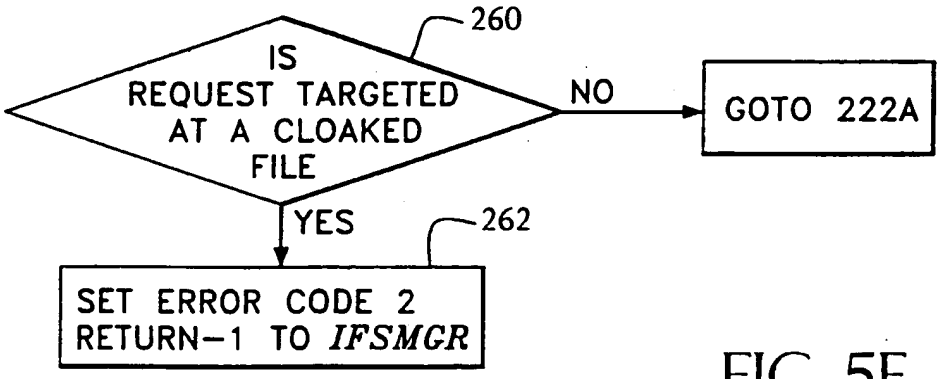


FIG. 5F

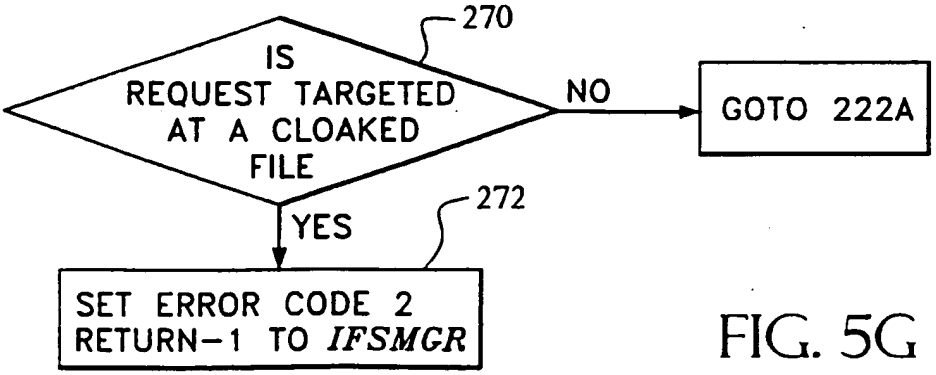


FIG. 5G

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/14055

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) : G06F 17/30

US CL : 707/1, 200, 201, 202, 203, 204, 205; 364/253

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 707/1, 200, 201, 202, 203, 204, 205; 364/253

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,809,230 A (PEREIRA) 15 September 1998, col. 4, lines 4-65, col. 5, lines 28-53, col. 7, lines 49-67, col. 8, lines 52-67, col. 9, lines 1-67, col. 10, lines 10-33, col. 11, lines 11-21, col. 12, lines 59-67, and col. 13, lines 1-3.	1-24
Y	US 5,544,360 A (LEWAK et al) 06 August 1996, Col. 7, lines 33-67, col. 8, lines 1-15 and lines 39-60, col. 9, lines 36-49, col. 11, lines 24-67, col. 12, lines 1-20, col. 13, lines 38-52, col. 14, lines 40-67, col. 15, lines 1-12 and lines 56-67, col. 16, lines 1-10 and lines 28-40, and col. 18, lines 61-67.	1-24

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 10 JULY 2000	Date of mailing of the international search report 01 AUG 2000
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer KIM VU Telephone No. (703) 305-4393

Form PCT/ISA/210 (second sheet) (July 1998)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/14055

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,832,527 A (KAVAGUCHI) 03 November 1998, col. 2, lines 25-45, col. 4, lines 15-54, col. 5, lines 8-67, col. 6, lines 1-5 and lines 56-67, col. 8, lines 1-20, col. 11, lines 47-53, col. 12, lines 17-40, col. 13, lines 15-26, and col. 18, lines 41-45.	1-24

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/14055

B. FIELDS SEARCHED
Electronic data bases consulted (Name of data base and where practicable terms used):

WEST
Search terms: file management, hidding file, file encryption, file manager, file deletion,security, secure file, conceal file, call processing, call system, file system.